



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра математического обеспечения и стандартизации информационных
технологий (МОСИТ)

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 7.2
по дисциплине
«Структуры и алгоритмы обработки данных»

Тема: «Графы: создание, алгоритмы обхода, важные задачи теории графов»

Выполнил студент: Лазаренко С.А.

Группа: ИКБО-10-23

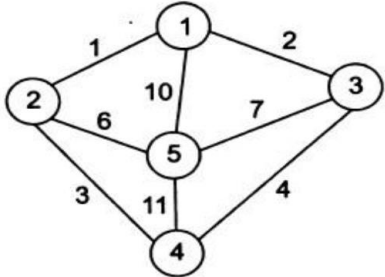
Москва 2024

СОДЕРЖАНИЕ

ЦЕЛЬ РАБОТЫ	3
ХОД РАБОТЫ	4
Формулировка задачи	4
Описание подхода к решению	4
Коды программы	4
Результаты тестирования	6
ВЫВОД.....	7
СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ	8

ЦЕЛЬ РАБОТЫ

Разработать алгоритм построения остовного дерева, указанный в варианте и реализовать программу.

17	Построение остовного дерева алгоритмом Крускала	
----	---	--

Провести тестовый прогон программы на предложенном в индивидуальном варианте задания графе.

Оформить отчет в соответствии с требованиями документирования.

ХОД РАБОТЫ

Формулировка задачи

Задача алгоритма Крускала состоит в том, чтобы найти минимальное остовное дерево для граф.

Индивидуальный вариант работы – 17.

Описание подхода к решению

Начинаем с того, что сортируем все рёбра графа по возрастанию их весов чтобы сначала рассматривать самые "лёгкие" рёбра, которые минимизируют общий вес будущего остовного дерева.

Проходя по отсортированному списку рёбер, добавляем каждое ребро, которое соединяет вершины из разных компонент связности. После добавления ребра компоненты этих вершин объединяются. Таким образом каждое добавленное ребро приближает нас к остовному дереву, не создавая циклов.

Код программы

Реализуем код программы на языке программирования C++ :

```
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

struct Edge {
    int src, dest, weight;
    Edge(int s, int d, int w) : src(s), dest(d), weight(w) {}
};

bool compare(Edge a, Edge b) {
    return a.weight < b.weight;
}

class Graph {
    int V;
    vector<Edge> edges;
public:
    Graph(int V) : V(V) {}

    void addEdge(int src, int dest, int weight) {
        edges.emplace_back(src, dest, weight);
    }

    int find(int parent[], int i) {
        if (parent[i] != i)
            parent[i] = find(parent, parent[i]);
        return parent[i];
    }
};
```

Рисунок 1 – код программы (1 часть)

```
void unionSets(int parent[], int rank[], int x, int y) {
    int rootX = find(parent, x);
    int rootY = find(parent, y);

    if (rank[rootX] < rank[rootY])
        parent[rootX] = rootY;
    else if (rank[rootX] > rank[rootY])
        parent[rootY] = rootX;
    else {
        parent[rootY] = rootX;
        rank[rootX]++;
    }
}

void kruskalMST() {
    sort(edges.begin(), edges.end(), compare);

    vector<Edge> result;
    int parent[V + 1];
    int rank[V + 1];

    for (int i = 1; i <= V; ++i) {
        parent[i] = i;
        rank[i] = 0;
    }

    for (Edge& edge : edges) {
        int root1 = find(parent, edge.src);
        int root2 = find(parent, edge.dest);

        if (root1 != root2) {
            result.push_back(edge);
            unionSets(parent, rank, root1, root2);
        }
    }
}
```

Рисунок 2 – Код программы (2 часть)

```
    cout << "Минимальное остовное дерево:\n";
    for (const Edge& e : result) {
        cout << e.src << " - " << e.dest << " : " << e.weight << endl;
    }
};

int main() {
    int V = 5;
    Graph graph(V);

    graph.addEdge(1, 2, 1);
    graph.addEdge(1, 3, 2);
    graph.addEdge(1, 5, 10);
    graph.addEdge(2, 5, 6);
    graph.addEdge(2, 4, 3);
    graph.addEdge(3, 5, 7);
    graph.addEdge(3, 4, 4);
    graph.addEdge(4, 5, 11);

    graph.kruskalMST();

    return 0;
}
```

Рисунок 3 – Код программы (3 часть)

Результаты тестирования

Выполним тестирование программы:

```
gwynbleidd@MacBook-Air-Sergej-3 практика 7.2 % ./prac7_2
Минимальное остовное дерево:
1 - 2 : 1
1 - 3 : 2
2 - 4 : 3
2 - 5 : 6
```

Рисунок 4 – Тестирование программы

Тестирование показало, что программа работает корректно.

ВЫВОД

В ходе работы была реализована программа для построения минимального остовного дерева графа методом Крускала. Метод Крускала позволяет эффективно решать задачу минимизации суммарного веса рёбер в остовном дереве графа, применяя жадный подход и упорядочивая рёбра по весам.

На этапе тестирования программа корректно выполнила построение минимального остовного дерева на предложенном графе, что подтверждает правильность её работы. В результате был получен минимальный набор рёбер, соединяющий все вершины графа без циклов и с минимальной суммарной стоимостью.

СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

1. Страуструп Б. Программирование. Принципы и практика с использованием C++. 2-е изд., 2016.
2. Документация по языку C++ [Электронный ресурс]. URL: <https://docs.microsoft.com/ru-ru/cpp/cpp/> (дата обращения 08.09.2024).
3. Курс: Структуры и алгоритмы обработки данных. Часть 2 [Электронный ресурс]. <https://online-edu.mirea.ru/course/view.php?id=4020> (дата обращения 04.09.2024)