Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный** **исследовательский политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

направление подготовки: 09.03.01 – «Информатика, вычислительная техника»

**О Т Ч Е Т**

**по лабораторной работе №2 на тему:  
 «Алгоритм Краскала»**

Выполнил студент гр. ИВТ-23-1б

Пискунов Дмитрий Александрович

(Фамилия, Имя, Отчество)

(подпись)

Проверил:

ст. преп. каф. ИТАС

Рустамханова Гульшат Ильдаровна

(должность, Ф.И.О руководителя от кафедры)

(оценка) (подпись)

(дата)

Пермь 2024

**Цель работы**

Исследование и программная реализация алгоритма Краскала для построения минимального остовного дерева графа.

**Задачи**

1. Разработать программу для построения минимального остовного дерева (MST) с использованием алгоритма Краскала.
2. Реализовать структуру данных Union-Find (система непересекающихся множеств) для эффективного управления компонентами связности.
3. Протестировать программу на примерах графов, заданных матрицей смежности.

**Теоретические сведения**

Минимальное остовное дерево (MST) — это подграф связного неориентированного взвешенного графа, который:

* Содержит все вершины исходного графа.
* Является деревом (без циклов).
* Имеет минимальную возможную сумму весов рёбер.

Алгоритм Краскала:

1. Сортировка рёбер графа по возрастанию веса.
2. Последовательное добавление рёбер в MST, начиная с самых лёгких, при условии, что они не создают циклы.
3. Использование структуры Union-Find для проверки наличия циклов:
   * **find(v)** — определяет корень компоненты, к которой принадлежит вершина **v**.
   * **union(a, b)** — объединяет компоненты вершин **a** и **b**.

Ключевые идеи:

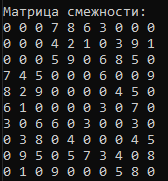
* Жадный подход: выбираются самые дешёвые рёбра, пока не будет построено дерево.
* Эффективность: время работы алгоритма — *O*(*E*log*E*), где *E* — количество рёбер.

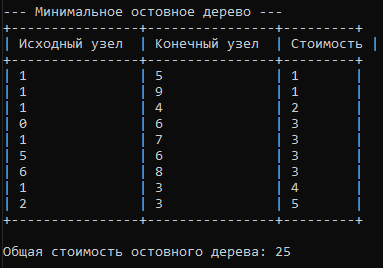
**Реализация программы**

1. Чтение матрицы смежности:
   * Матрица считывается из файла (например, **graph.txt**).
   * Размер матрицы фиксирован: **MAX\_NODES = 10**.
   * Ненулевые значения интерпретируются как веса рёбер.
2. Преобразование в список рёбер:
   * Рёбра извлекаются из верхнего треугольника матрицы, чтобы избежать дублирования.
   * Каждое ребро хранится в структуре **Edge** с полями **from**, **to**, **cost**.
3. Сортировка рёбер:
   * Рёбра сортируются по возрастанию веса с использованием **std::sort**.
4. Структура Union-Find:
   * Вектор **parent** хранит родителей вершин.
   * Вектор **rank** используется для балансировки деревьев (оптимизация времени поиска).
5. Построение MST:
   * Для каждого ребра проверяется, принадлежат ли его вершины разным компонентам.
   * Если да, ребро добавляется в MST, а компоненты объединяются.

**Пример работы программы**

Входная матрица (файл g21.txt):

****  
Вывод программы:

****

**Заключение**

Программа корректно реализует алгоритм Краскала для построения MST.

Использование структуры Union-Find обеспечивает эффективное управление компонентами связности.

Ограничения программы:

1) Фиксированный размер матрицы (**MAX\_NODES = 10**).

2) Отсутствие проверки корректности входных данных (например, несимметричная матрица).

Код программы представлен на GitHub по адресу: https://github.com/MOkASiH/Math/tree/main/2\_sem/2