

Теория графов. Первая презентация

Шмаков Александр



Введение

Остовное дерево — дерево, подграф исходного графа, имеющий такое же число вершин.

Минимальное остовное дерево — остовное дерево взвешенного графа, имеющее минимальный возможный вес.

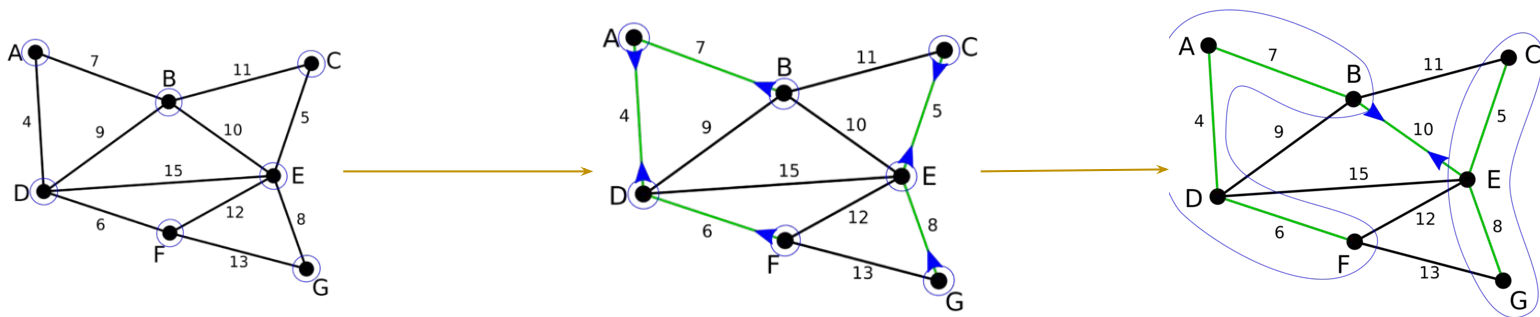
Алгоритм Борувки — алгоритм нахождения минимального остовного дерева в графе.

Алгоритм Борувки

ВХОД: Взвешенный граф $G = (V, E)$.

ВЫХОД: Подграф F , минимальное остовное дерево.

1. Каждая вершина графа G — дерево, никакие ребра им не принадлежат.
2. Каждому дереву добавим минимальное инцидентное ему ребро.
3. Повторяем предыдущий шаг, пока не останется только одна компонента.



Характеристики вычислительной машины

- **Процессор:** Intel Core Ultra 9 185H (16 ядер, 22 потоков)
 - Базовая частота: 2300MHz
 - L1-кэш: 672 KB
 - L2-кэш: 12 МБ
 - L3-кэш: 24 МБ
- **RAM:** 32 GB
- **GPU:** NVIDIA GeForce RTX 4060 Max-Q / Mobile
 - CUDA 12.4
 - 8 Gb VRAM
- **Операционная система:** Ubuntu 25.10

Dataset DIMACS 9th

Description	Nodes	Edges
Great Lakes	2,758,119	3,397,404
California and Nevada	1,890,815	2,315,222
Northeast USA	1,524,453	1,934,010
Northwest USA	1,207,945	1,410,387
Florida	1,070,376	1,343,951
Colorado	435,666	521,200
San Francisco Bay Area	321,270	397,415
New York City	264,346	365,050

Эксперимент

- На какой из двух библиотек (**Gunrock**, **Apache Giraph**) быстрее реализация алгоритма Борувки?
- Конкретная реализация запускается на каждом графе 20 раз.

Реализация на Gunrock

Gunrock

Gunrock — это высокопроизводительная библиотека для обработки графов на GPU с использованием CUDA, которая упрощает разработку параллельных графовых алгоритмов за счет встроенных в библиотеку готовых механизмов.

Шаги:

1. Загрузка графа
2. Каждая вершина – отдельная компонента
3. Фильтруем (**filter**) ребра, выбираем минимальное для каждой компоненты
4. Объединяем компоненты
5. Повторяем, пока не останется 1 компонента

Термины и операции:

1. **advance** – создает новый фронт из текущего путем обхода соседних элементов, работая либо с вершинами, либо с ребрами
2. **filter** – создает новый фронт из текущего путем выбора подмножества текущего фронта на основе заданного критерия
3. **MST** – Minimum Spanning Tree

Реализация на Apache Giraph

Apache Giraph

Apache Giraph — распределённая платформа для обработки графов по модели Pregel: каждая вершина параллельно выполняет обработку, обменивается сообщениями, а глобальная синхронизация происходит по *суперстепам*.

Шаги:

1. Загрузка графа.
2. Суперстеп «*рассылка*»: вершины отправляют соседям свой `componentId`.
3. Суперстеп «*выбор кандидатов*»: каждая вершина просматривает исходящие рёбра, выбирает минимальные и посылает предложения (`candidate`) корню своей компоненты или агрегатору.
4. Суперстеп «*глобальный выбор*»: фиксируется по одному минимальному ребру на компоненту.
5. Суперстеп «*слияние*»: вершины обновляют `componentId`.
6. Повторяем суперстепы 2–5, пока число компонент не станет равно 1.