Лабораторная работа №9 по курсу дискретного анализа: Графы

Выполнил студент группы М8О-312Б-22 МАИ *Корнев Максим*.

Условие

Вариант: 6. Поиск кратчайших путей между всеми парами вершин алгоритмом Джонсона

Задан взвешенный ориентированный граф, состоящий из n вершин и m ребер. Вершины пронумерованы целыми числами от 1 до n. Необходимо найти длины кратчайших путей между всеми парами вершин при помощи алгоритма Джонсона. Длина пути равна сумме весов ребер на этом пути. Обратите внимание, что в данном варианте веса ребер могут быть отрицательными, поскольку алгоритм умеет с ними работать. Граф не содержит петель и кратных ребер.

Метод решения

1. Так как алгоритм Дейкстры не умеет работать с отрицательными рёбрами, необходимо на время избавиться от них в нашем графе. Для этого мы добавляем в граф фиктивную вершину S и строим из неё рёбра с весом 0 в каждую вершину исходного графа.

2. Для нового графа запускаем алгоритм Беллмана – Форда, который либо обнаруживает наличие отрицательного цикла в графе и завершает алгоритм, либо возвращает кратчайшие расстояния от фиктивной вершины S до каждой вершины исходного графа. Суть алгоритма заключается в том, что мы $V-1$ раз проходим по всем рёбрам и релаксируем их, если

$$d[v] > d[u] + w(u, w).$$

   Если на V-ой итерации происходит ещё одна релаксация, то в графе имеется отрицательный цикл. С помощью этих кратчайших расстояний мы перевзвешиваем рёбра по следующей формуле:

$$\omega\varphi(u, v) = \omega(u, v) + \varphi(u) - \varphi(v).$$

   Удаляем фиктивную вершину и запускаем алгоритм Дейкстры для каждой вершины графа, который возвращает кратчайшие расстояния до каждой другой вершины графа. Для преобразования этих расстояний к изначальному графу необходимо применить обратную формулу перевзвешивания:

$$\omega\varphi(u, v) = \omega(u, v) - \varphi(u) + \varphi(v).$$

3. Суть алгоритма Дейкстры заключается в том, что в алгоритме поддерживается множество вершин, для которых уже вычислены длины кратчайших путей до них из s. На каждой итерации основного цикла выбирается вершина, не помеченная посещённой, которой на текущий момент соответствует минимальная оценка кратчайшего пути. Вершина добавляется в множество посещённых и производится релаксация всех исходящих из неё рёбер.

Описание программы

Для реализации алгоритма были реализованы следующие функции:

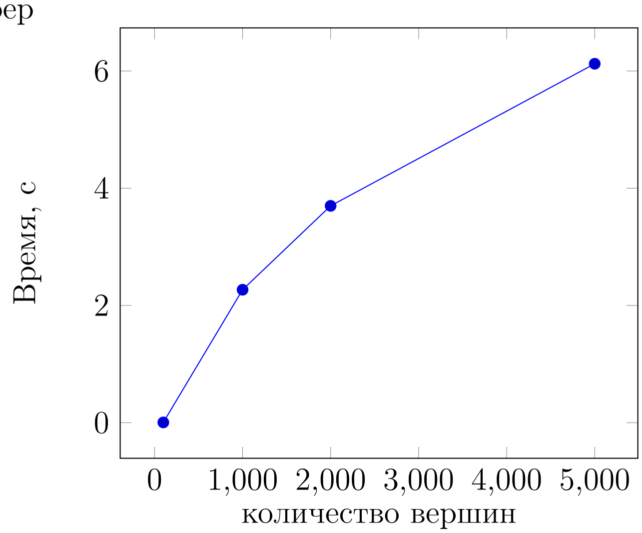
1. **BellmanFord(Graph* graph)**:
   - Выполняет алгоритм Беллмана-Форда для поиска кратчайших расстояний до всех вершин графа.
   - Проверяет наличие отрицательных циклов в графе:
   $$d[from] + cost < d[to],.$$
   - Перевзвешивает рёбра, корректируя их стоимости с учётом расстояний.
2. **Dijkstra(const Graph* graph, const long long* baseDistances)**:
   - Для каждой вершины выполняет алгоритм Дейкстры, начиная с неё как стартовой.
   - Использует массив расстояний и массив посещённых вершин для поиска кратчайших путей.
   - В каждой итерации выбирает вершину с минимальным текущим расстоянием.
   - Обновляет расстояния для соседних вершин через рёбра графа.
   - Корректирует финальные расстояния с учётом базовых расстояний:
   $$d[i] = distances[i] - baseDistances[startVertex] + baseDistances[i].$$

Дневник отладки

1. Получил WA на 1 тесте, так как перепутал знаки в формуле Беллмана-Форда.

Тест производительности

Алгоритм работает за время $O(V^3 + VE)$, где V - количество вершин E - количество ребер



Выводы

В ходе выполнения лабораторной работы я изучил алгоритм Джонсона и применил его для решения задачи нахождения кратчайших путей между всеми парами вершин в графе. Полученные знания позволили успешно реализовать и проверить работу алгоритма на практике.