Министерство образования и науки Украины

ГВУЗ «Приазовский государственный технический университет»

Кафедра компьютерных наук

**Реферат**

по дисциплине «Информационные системы и технологии»

**Тема работы: «Алгоритм Беллмана-Форда»**

Выполнил

студент группы КН-18

Шевчук Руслан Дмитриевич

Приняла

ст. преп. Чемова А.В.

Мариуполь, 2019

**Оглавление**

1. **Описание алгоритма (общее и математическое)**
2. **Формулировка задачи**
3. **Схема реализации в виде псевдокода**
4. **1** **Пример графа с весовой матрицей**
5. **2** **Пример кода на C++ по заданному графу**
6. **Выводы**
7. **Описание алгоритма**

**Общее:**

Алгоритм Беллмана-Форда предназначен для решения задачи поиска кратчайшего пути на графе. Для заданного ориентированного взвешенного графа алгоритм находит кратчайшие расстояния от выделенной вершины-источника до всех остальных вершин графа. Алгоритм Беллмана-Форда масштабируется хуже других алгоритмов решения указанной задачи, однако его отличительной особенностью является применимость к графам с произвольными, в том числе отрицательными, весами.

**Математическое:**

Пусть задан граф G = (V, E) с весами рёбер f(e) и выделенной вершиной источником u. Обозначим через d(v) кратчайшее расстояние от источника u

до вершины v.

Алгоритм Беллмана-Форда ищет функцию d(v) как единственное решение уравнения

d(v)=min{d(w) + f(e) ∣ e = (w, v) ∈ E},

∀v ≠ u,

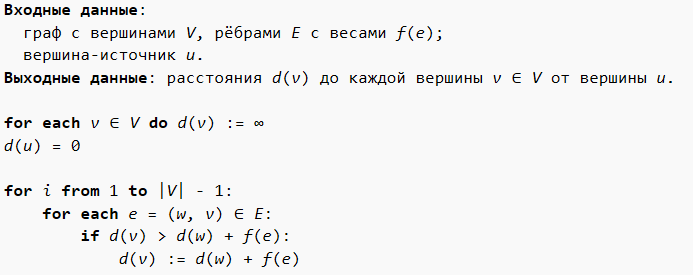
с начальным условием d(u) = 0.

1. **Формулировка задачи**

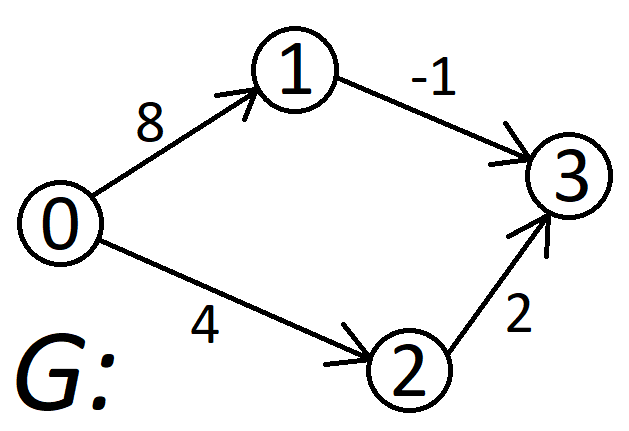
Дан ориентированный или неориентированный граф G со взвешенными рёбрами. Длиной пути назовём сумму весов рёбер, входящих в этот путь. Требуется найти кратчайшие пути от выделенной вершины s до всех вершин графа.

Заметим, что кратчайших путей может не существовать. Так, в графе, содержащем цикл с отрицательным суммарным весом, существует сколь угодно короткий путь от одной вершины этого цикла до другой (каждый обход цикла уменьшает длину пути). Цикл, сумма весов рёбер которого отрицательна, называется отрицательным циклом.

1. **Схема реализации в виде псевдокода**

****

1. **1 Пример графа с весовой матрицей**

****

1. **2 Пример кода на C++ по заданному графу**

**Код из файла “bellman\_ford.h”**

#include <cstdio>

#include <cstdlib>

#include <climits>

struct Edge

{

int src, dest, weight;

};

struct Graph

{

int V, E;

struct Edge\* edge;

};

struct Graph\* createGraph(int V, int E)

{

struct Graph\* graph = new Graph;

graph->V = V;

graph->E = E;

graph->edge = new Edge[E];

return graph;

}

void printArr(int dist[], int n, int source)

{

printf("Source vertex: %d\n", source);

printf("Vertex Distance from Source\n");

for (int i = 0; i < n; ++i)

{

if(dist[i]==INT\_MAX) printf("%d \t\t Unreachable\n", i);

else printf("%d \t\t %d\n", i, dist[i]);

}

}

void Bellman\_Ford(struct Graph\* graph, int src)

{

int V = graph->V;

int E = graph->E;

int dist[V];

for (int i = 0; i < V; i++)

dist[i] = INT\_MAX;

dist[src] = 0;

for (int i = 1; i <= V - 1; i++)

{

for (int j = 0; j < E; j++)

{

int u = graph->edge[j].src;

int v = graph->edge[j].dest;

int weight = graph->edge[j].weight;

if (dist[u] != INT\_MAX && dist[u] + weight < dist[v])

dist[v] = dist[u] + weight;

}

}

for (int i = 0; i < E; i++)

{

int u = graph->edge[i].src;

int v = graph->edge[i].dest;

int weight = graph->edge[i].weight;

if (dist[u] != INT\_MAX && dist[u] + weight < dist[v])

printf("Graph contains negative weight cycle");

}

printArr(dist, V, src);

return;

}

**Код из файла “main.cpp”**

#include "bellman\_ford.h"

int main(void)

{

int V = 4;

int E = 4;

struct Graph\* graph = createGraph(V, E);

graph->edge[0].src = 0;

graph->edge[0].dest = 1;

graph->edge[0].weight = 8;

graph->edge[1].src = 0;

graph->edge[1].dest = 2;

graph->edge[1].weight = 4;

graph->edge[2].src = 2;

graph->edge[2].dest = 3;

graph->edge[2].weight = 2;

graph->edge[3].src = 1;

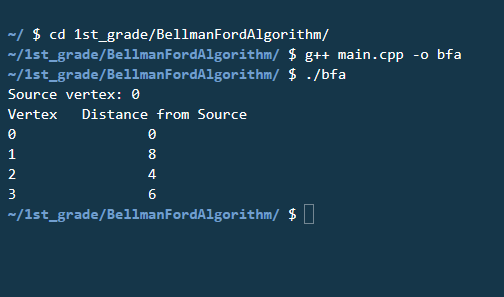
graph->edge[3].dest = 3;

graph->edge[3].weight = -1;

Bellman\_Ford(graph, 0);

return 0;

}



1. **Вывод**

*Алгоритм Беллмана-Форда предназначен для решения задачи поиска кратчайшего пути на графе; он занимает больше времени на итерации чем алгоритм Дейкстры, но его особенной характеристикой является то, что с помощью него можно искать кратчайший путь у ребер с отрицательным весом.*