Минестерство образования и науки Российской Федерации

ФГБОУ ВО Ульяновский государственный технический университет

Кафедра «вычислительная техника»

Лабораторная работа №4

«Нахождение кратчайшего пути в графе»

Выполнил студент

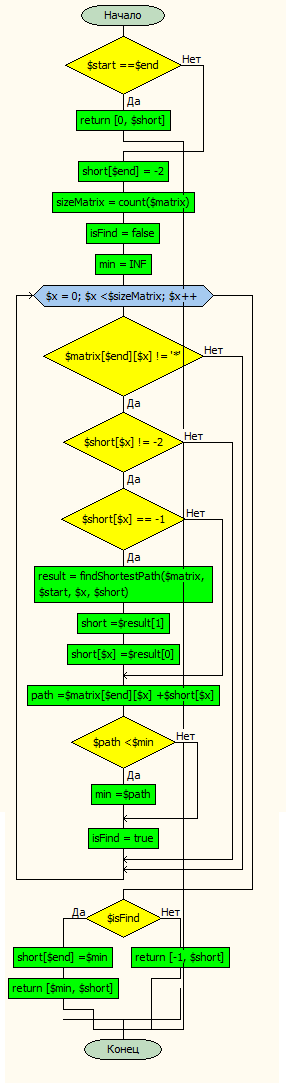
Группы ИВТАСбд-11

Долгов Д. П.

Ульяновск, 2022

**Работа №4. Нахождение кратчайшего пути в графе**

**Цель работы.**

Требуется написать программу, которая будет находить кратчайший путь между двумя узлами в графе. Основной алгоритм должен быть реализован на языке программирования php, обязательно должна присутствовать проверка корректности ввода данных. Пользователь обладает возможностью ввода данных.

**Описание метода выполнения**

Алгоритм нахождения кратчайшего пути в графе реализован с помощью рекурсивной функции, которая находит кратчайшие пути для каждого из узлов графа.

//$start, $end - просто числа, которые содержат номера точек

//$short - массив, содержащий кратчайшие длины пути до точки. С самого начала должен быть заполнен -1

function findShortestPath($matrix, $start, $end, $short)

{

if($start == $end)

return [0, $short];

$short[$end] = -2;

$sizeMatrix = count($matrix);

$isFind = false;

$min = *INF*;

for($x = 0; $x < $sizeMatrix; $x++)

{

if($matrix[$end][$x] != '\*') {

if($short[$x] != -2) {

if ($short[$x] == -1) {

$result = findShortestPath($matrix, $start, $x, $short);

$short = $result[1];

$short[$x] = $result[0];

}

$path = $matrix[$end][$x] + $short[$x];

if ($path < $min) $min = $path;

$isFind = true;

}

}

}

if($isFind) {

$short[$end] = $min;

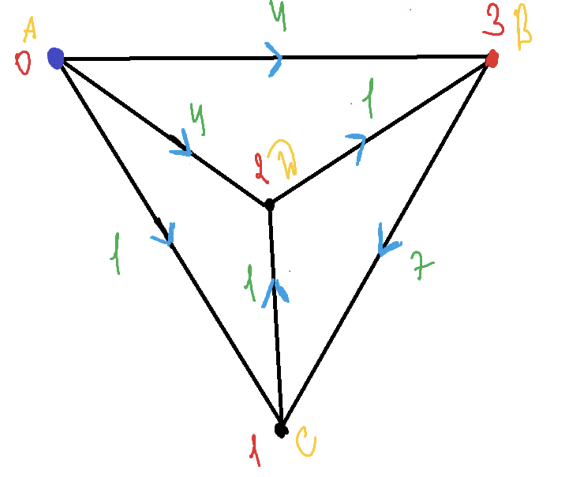
return [$min, $short];

} else{

return [-1, $short];

}

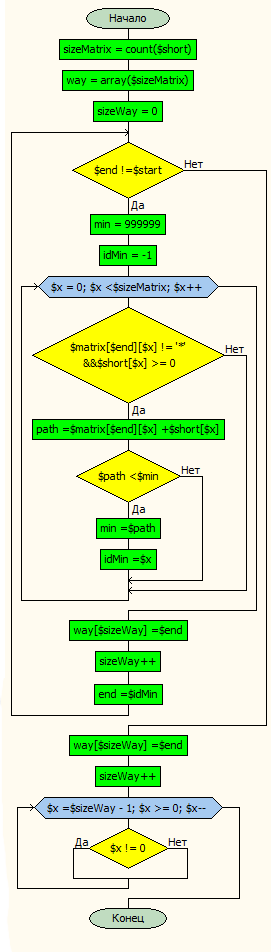
}



Принцип работы алгоритма следующий. Допустим, требуется найти кратчайшее расстояние на графе от точка A до точки B (пример графа на изображении). Алгоритм начинает свою работу с конца, то есть с самого начала в рассмотрения берётся последняя точная, то есть точно B. Для того, чтобы определить кратчайшее расстояние до этой точки от точки A, можно перебрать все узлы, которые входят в B, взять кратчайшие расстояние до них от точка A и сложить с длиной ребра, соединяющего данную точку с точкой B. Из полученных расстояние остаётся лишь выбрать минимальное.

Но самое главное здесь то, что кратчайшее расстояние до точек, которые соединяются с узлом B неизвестно, но его можно получить с помощью использования той же самой функции, которая определяет кратчайшее расстояние до точки B. Таким образом получается вызов функции самой себя или рекурсия.

Эта рекурсивная функция должна вернуть значение 0, как только она доберется до точки старта, так как расстояние от точка старта до точки старта равно нулю.

Также, чтобы не просчитывать кратчайшие расстояние до отдельных узлов повторно, можно записывать уже просчитанные расстояние в отдельные массив - именно с помощью него можно определить узлы, составляющие минимальный путь, а затем вывести эту последовательность узлов на страницу сайта. Вот так выглядит функция, которая это делает:

function findPath($matrix, $short, $start, $end)

{

$sizeMatrix = count($short);

$way = array($sizeMatrix);

$sizeWay = 0;

while($end != $start) {

$min = 999999;

$idMin = -1;

for ($x = 0; $x < $sizeMatrix; $x++) {

if ($matrix[$end][$x] != '\*' && $short[$x] >= 0) {

$path = $matrix[$end][$x] + $short[$x];

if ($path < $min) {

$min = $path;

$idMin = $x;

}

}

}

$way[$sizeWay] = $end;

$sizeWay++;

$end = $idMin;

}

$way[$sizeWay] = $end;

$sizeWay++;

for($x = $sizeWay - 1; $x >= 0; $x--){

echo $way[$x];

if($x != 0)

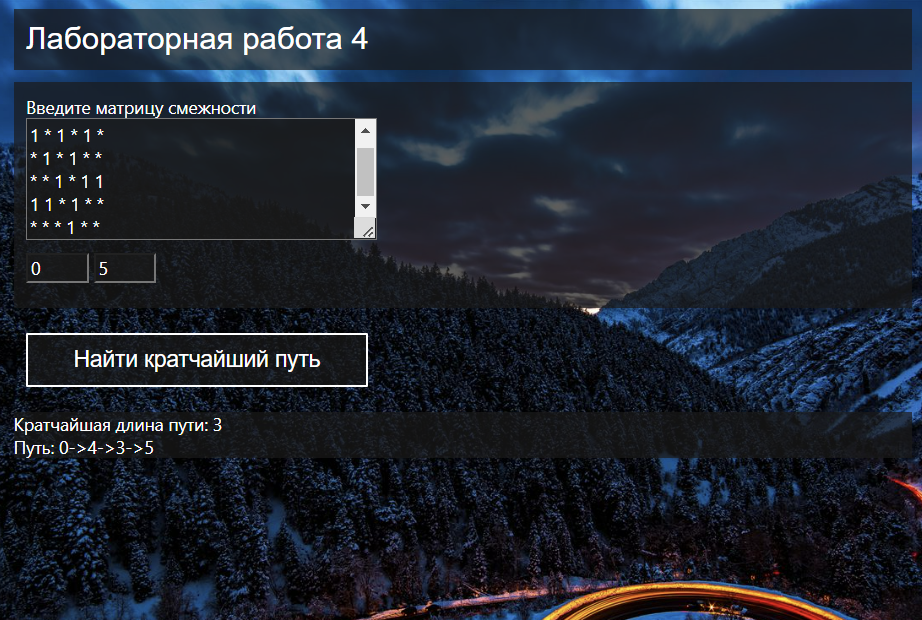
echo "->";

}

}

Здесь по-сути реализуется тот же алгоритм нахождения кратчайшего пути, но с учетом того, что все кратчайшие расстояния для каждой из точек уже известны. В итоге, его реализация возможно без рекурсии.

Интерфейс для работы с данным алгоритм выглядит следующим образом:



Здесь присутствует заголовок - номер лабораторной работы, поле для ввода матрицы смежности, два поля для ввода начальной и конечной точки, между которыми находится кратчайшее расстояние, а также кнопка, по нажатию на которую выводится длина кратчайшего пути, а также последовательность узлов из которых состоит кратчайший путь. Матрица смежности вводится в виде натуральных чисел и звездочек, где звездочка - обозначение, что пути нет, натуральное число - длина дуги.

В интерфейсе также присутствуют стили оформления элементов интерфейса, а также фоновое изображение.

Чтобы по нажатию на кнопку можно было передать информацию в код, написанный на php, используется ajax запрос на языке программирования javaScript. Код для вызова ajax запроса находится в библиотеки JQuery. Вызов самой функции и передача данные происходит следующим образом:

function send(message)

{

$.ajax({

type: "POST",

url: 'scripts/matrix.php',

data: message,

success: function(data){

***document***.getElementById("result").innerHTML = data;

}

});

}

**Выводы**

Программа протестирована и работает успешно. Эта программа верно выполняет поиск кратчайшего пути в графе, а также корректно отображает его на сайте.

**Список литературы**

Графы, вершины, ребра, инцидентность, смежность [2020]. [Электронный ресурс] URL: <https://www.youtube.com/watch?v=fCfPjm8u89U>

Графы [Электронный ресурс] URL: <https://www.youtube.com/watch?v=9fdkrSvgqeo>