Министерство образования и науки Российской Федерации

Ульяновский Технический университет

Кафедра: вычислительная техника

Дисциплина: Дискретная математика

Лабораторная работа №4.

«Графы»

Выполнил:

Студент: ИВТАП Бд-11

Кондратьев Павел Сергеевич

Проверил:

Игонин Андрей Геннадьевич

Ульяновск, 2017

Содержание

|  |  |
| --- | --- |
| 1. Цели и задачи……………………………………………………………… | 3 |
| 2. Ход выполнения работы…………………………………………………. | 3 |
| 3. Выводы ……………………………………………………………………. | 5 |
| 4. Список используемой литературы…...…………………………………... | 5 |
| 5. Приложение №1…………………………………………………………… | 6 |

**Цели и задачи**

**Цель**

Развитие, расширение и понимание устройства Интернета, с помощью изучения языков программирования, которые используются в создании сайта. Изучение языка программирования PHP и его использование в различных задачах.

**Задача**

Разработать программу на php выполняющую поиск самого короткого пути (цепи) между двумя точками (вершинами) на графе, в которой минимизируется сумма весов рёбер, составляющих путь. Ввод графа (матрица весов) осуществляется пользователем в отведенном для этого месте (Форма ввода данных). Программа ищет кратчайший путь из А в Б (вершина А и Б задаются в форме) при этом программа выводит какие вершины были посещены, также производиться ввод количества вершин.

**Ход выполнения работы**

Для реализации данной задачи нам потребуется:

1. Создать форму для ввода данных
2. Разработать алгоритм нахождения кратчайшего пути в графе
3. Реализовать вывод посещенных вершин

Для создания формы ввода, потребуется 4 тега <input> в самой <form>, 3 из которых отвечают за ввод данных (для этого указываем: type="text") и 4 <input> отвечает за действие над множествами и передаем ему type="submit"( Кнопка для отправки данных формы на сервер). Также создаем textarea для ввода матрицы весов графа.

<form method="post" action="3.php">  
 Graf <br>  
 <input name="size" type="text" placeholder="n = Колличество вершин"><br>Инициализация вершин начинается с 0 и до n - 1<br>  
 <input name="top" size="25" type="text" placeholder="В какой верщине мы находимся?">**<?php echo** "a = ", $\_POST['top']; **?>** <br>  
 <input name="next" size="25" type="text" placeholder="В какую верщину мы идем?">**<?php echo** "b = ", $\_POST['next']; **?>** <br><br>  
 Инициализация матрицы [**<?=** $\_POST['size'] **?>**][**<?=** $\_POST['size'] **?>**](весов графа)  
 <p><textarea name="links" rows="9" cols="35"  
 style="resize: none;">**<?=** htmlspecialchars($\_POST['links']) **?>**</textarea></p>  
 <p><input type="submit" name="button" value="сделать"/></p>  
</form>

В данной программе использовался алгоритм Дейкстры - алгоритм на графах, изобретённый нидерландским ученым Э. Дейкстрой в 1959 году. Находит кратчайшее расстояние от одной из вершин графа до всех остальных.

Перво-наперво нужно инициализация вершины и расстояния (задать минимальное расстояние до вершин-бесконечность, заполнить не посещённые вершины 0 и заполнить матрицу предком (-1) для вывода Пути). Дальше идет шаг алгоритма. Каждой вершине из V сопоставим метку - минимальное известное расстояние от этой вершины до А. Алгоритм работает пошагово - на каждом шаге он «посещает» одну вершину и пытается уменьшать метки. Работа алгоритма завершается, когда все вершины посещены.

В противном случае, из ещё не посещённых вершин выбирается вершина U, имеющая минимальную метку. Мы рассматриваем всевозможные маршруты, в которых U является предпоследним пунктом. Вершины, в которые ведут рёбра из U, назовём соседями этой вершины. Для каждого соседа вершины U, кроме отмеченных как посещённые, рассмотрим новую длину пути, равную сумме значений текущей метки U и длины ребра, соединяющего U с этим соседом. Если полученное значение длины меньше значения метки соседа, заменим значение метки полученным значением длины. Рассмотрев всех соседей, пометим вершину U как посещённую и повторим шаг алгоритма.

**do** {  
 $minindex=10000;  
 $min=10000;  
 **for**($i=0;$i<*SIZE*;$i++){  
 **if** ($v[$i]==0 && $d[$i]<$min){  
 $min=$d[$i];  
 $minindex=$i;  
 }  
 }  
 // Добавляем найденный минимальный веск текущему весу вершины и сравниваем с текущим минимальным весом вершины  
 **if**($minindex != 10000){  
 **for**($i=0;$i<*SIZE*;$i++){  
 **if** ($a[$minindex][$i] > 0) {  
 $temp = $min + $a[$minindex][$i];  
 //echo $a[$minindex][$i]," ", $min,"|";  
 **if** ($temp < $d[$i]){  
 $d[$i] = $temp;  
 $prev[$i]=$minindex;//Запоминаем предка посещенного города  
 }  
 }  
 }  
 $v[$minindex]=1;  
 }  
} **while** ($minindex<10000);

Для вывода пути по которому ы шли мы запоминаем предка посещенного города ($prev[$i]=$minindex;). После чего в цикле относительно введенного города выводим пути и кратчайший путь графа.

**for**($z=*Next* - 1;$z<*Next*;$z++){  
 **if**($d[$z]!=10000){  
 **echo** "Расстояние из города ";  
 **echo** *Top*+1 ." до города ";  
 **echo** $z+1 ." = ";  
 **echo** $d[$z];  
 **echo** " Путь : ";  
  
 $finalpath = **array**();  
 $prev[*Top*]=*Top*;  
 $k=0;  
  
 $index=$z;  
  
 **echo** *Top*+1;  
 **echo** " => ";  
 //echo "|",$prev[$index],"|";  
 **while**($prev[$index]!= *Top*){  
 $finalpath[$k]=$prev[$index]+1;  
 $index=$prev[$index];  
 $k++;  
 }  
  
 **for**($j=count($finalpath)-1;$j>=0;$j--){  
 **echo** $finalpath[$j];  
 **echo** " => ";  
 }  
 **echo** $z+1;  
  
 **echo** "<br>";  
 }  
}

**Вывод**

Ознакомились с языком программирования - PHP и его использованием в различных задачах, также выполнили операции с графом. Изучили теорию графов и работу с ними. Были рассмотрены несколько алгоритмов нахождения кратчайшего пути в графе. Использовалась работа с матрицами, представления их в виде многомерного массива. Научились работать с пользователем через форму ввода данных на сайте.

**Список используемой литературы**

1. Алгоритм Дейкстры:[Электронный ресурс] URL: https://ru.wikipedia.org/wiki/Алгоритм\_Дейкстры
2. Лекции Игонин Андрей Геннадьевич по интернет программированию

**Приложение №1**

<form method="post" action="3.php">  
 Graf <br>  
 <input name="size" type="text" placeholder="n = Колличество вершин">  
 **<?php  
 if** (**isset**($\_POST['button'])) {  
 **if** (is\_numeric($\_POST['size'])) {  
 **echo** 'в поле содержится число</br>';  
 } **else** {  
 **die**();  
 }  
 }  
 **?>** <br>Инициализация вершин начинается с 0 и до n - 1<br>  
 <input name="top" size="25" type="text" placeholder="В какой верщине мы находимся?">**<?php echo** "a = ", $\_POST['top']; **?>** <br>  
 <input name="next" size="25" type="text" placeholder="В какую верщину мы идем?">**<?php echo** "b = ", $\_POST['next']; **?>** <br><br>  
 Инициализация матрицы [**<?=** $\_POST['size'] **?>**][**<?=** $\_POST['size'] **?>**](весов графа)  
 <p><textarea name="links" rows="9" cols="35"  
 style="resize: none;">**<?=** htmlspecialchars($\_POST['links']) **?>**</textarea></p>  
 <p><input type="submit" name="button" value="сделать"/></p>  
</form>  
**<?php** //echo "n = ", $\_POST['size']; **?>**<!--<br>-->  
  
**<?php  
if** (**isset**($\_POST['button'])) {  
 // echo "Квадратная матрица[", $\_POST['size'], "][", $\_POST['size'], "] для хранения весов графа", "<br/>";  
 define("SIZE", $\_POST['size'], **true**);  
 define("Top", $\_POST['top']-1, **true**);  
 define("Next", $\_POST['next'], **true**);  
 $b = preg\_split("/[\r, ]/", $\_POST['links']);  
 $a = array\_chunk($b, *SIZE*);  
 $d[*SIZE*]; // минимальное расстояние  
 $v[*SIZE*]; // посещенные вершины  
// Вывод матрицы связей  
// for ($i = 0; $i < SIZE; $i++) {  
// for ($j = 0; $j < SIZE; $j++) {  
// echo $a[$i][$j], " ";  
// }  
// echo "<br/>";  
// }  
 **if**(*Top*+1>*SIZE* || *Top*<0){ //Проверка ввода  
 **echo** "Такого города не существует";  
 **return** 0;  
 }  
//Инициализация вершин и расстояний  
 **for**($i=0;$i<*SIZE*;$i++){  
 $d[$i]=10000;  
 $v[$i]=0;  
 $prev[$i]=-1;//матрица предков  
 }  
 $d[*Top*]=0;  
  
  
// Шаг алгоритма  
 **do** {  
 $minindex=10000;  
 $min=10000;  
 **for**($i=0;$i<*SIZE*;$i++){  
 **if** ($v[$i]==0 && $d[$i]<$min){  
 $min=$d[$i];  
 $minindex=$i;  
 }  
 }  
 // Добавляем найденный минимальный веск текущему весу вершины и сравниваем с текущим минимальным весом вершины  
 **if**($minindex != 10000){  
 **for**($i=0;$i<*SIZE*;$i++){  
 **if** ($a[$minindex][$i] > 0) {  
 $temp = $min + $a[$minindex][$i];  
 //echo $a[$minindex][$i]," ", $min,"|";  
 **if** ($temp < $d[$i]){  
 $d[$i] = $temp;  
 $prev[$i]=$minindex;//Запоминаем предка посещенного города  
 }  
 }  
 }  
 $v[$minindex]=1;  
 }  
 } **while** ($minindex<10000);  
  
  
  
 **echo** "Длина пути из города ";  
 **echo** *Top*+1 . " до ", *Next*;  
 **echo** "<br><br>";  
 **for**($z=*Next* - 1;$z<*Next*;$z++){  
 **if**($d[$z]!=10000){  
 **echo** "Расстояние из города ";  
 **echo** *Top*+1 ." до города ";  
 **echo** $z+1 ." = ";  
 **echo** $d[$z];  
 **echo** " Путь : ";  
 $finalpath = **array**();  
 $prev[*Top*]=*Top*;  
 $k=0;  
 $index=$z;  
 **echo** *Top*+1;  
 **echo** " => ";  
 //echo "|",$prev[$index],"|";  
 **while**($prev[$index]!= *Top*){  
 $finalpath[$k]=$prev[$index]+1;  
 $index=$prev[$index];  
 $k++;  
 }  
  
 **for**($j=count($finalpath)-1;$j>=0;$j--){  
 **echo** $finalpath[$j];  
 **echo** " => ";  
 }  
 **echo** $z+1;  
 **echo** "<br>";  
 }  
 }  
}  
**?>**