Минестерство образования и науки Российской Федерации

ФГБОУ ВО Ульяновский государственный технический университет

Кафедра «вычислительная техника»

Лабораторная работа №5

«Нахождение матрицы достижимости»

Выполнил студент

Группы ИВТАСбд-11

Долгов А. П.

Ульяновск, 2022

**Цель работы:**

Разработать программу на php нахождения матрицы достижимости.

**Описание метода выполнения.**

Граф - это множество упорядоченных пар G := (V, E), где V – множество вершин графа, а E – множество рёбер графа. Оба множества непустые. Графы бывают ориентированные и неориентированные. Их отличие заключается в том, что в первом случае проход по ребру ориентированного графа (дуга) возможен только в одном направлении, а во втором – в обоих. Матрица смежности – квадратная матрица представления отношения вершин и рёбер, где значение элемента говорит о весе ребра из i точки в j. Вес ребра — значение, поставленное в соответствие данному ребру. Кратчайший путь в графе – последовательность вершин, соединённых рёбрами или дугами, у которых сумма весов этих рёбер или дуг наименьшая. Матрица достижимости – это такая, матрица в которой хранится информация о всех возможных путях из какой-либо вершины. По-другому – это бинарная матрица замыкания по транзитивности отношения.

Граф указывается через матрицу смежности. Ввод матрицы смежности осуществляется через специальное поле textarea. Под полем ввода находится кнопка «Get», на нажатию на которую будет производится расчёт матрицы достижимости. В случае некорректности введённых данных будет выведена ошибка.

В js файле реализован метод передачи сообщения в php файл:

function send(message)  
{  
 $.ajax({  
 url: 'php/shortestPath.php',  
 type: 'POST',  
 data: message,  
 success: function(data){  
 document.getElementById('divForOutput').innerHTML = data;  
 //('p.output').text('Data sent!');  
 }  
 })  
}

Само сообщение задаётся в html структуре в onclick поле кнопки.

<textarea id = "matrix" cols = "25" rows = "5"></textarea><br>  
<button id = "getButton" onclick = "  
 let matrix = document.getElementById('matrix').value;  
 let message = {'matrix': matrix};  
  
 send(message, 'php/script.php');  
">Get</button>

">Get shortest path</button><br><br>

В PHP файле первым делом происходит считывание данных:

$inputMatrix = $\_POST['matrix'];

Затем производится инициализация класса матрицы:

$matrix = new Matrix();  
$matrix->loadMatrix($inputMatrix);

Дальше происходит проверка введённых данных на ошибки, расчёт матрицы достижимости и вывод данных.

if($matrix->errorCode == 0)  
{  
 $resultMatrix = $matrix->getReachabilityMatrix();  
  
 if($resultMatrix->errorCode == 0)  
 {  
 $resultMatrix->printMatrix();  
 }else printError($resultMatrix->errorCode);  
}  
else printError($matrix->errorCode);

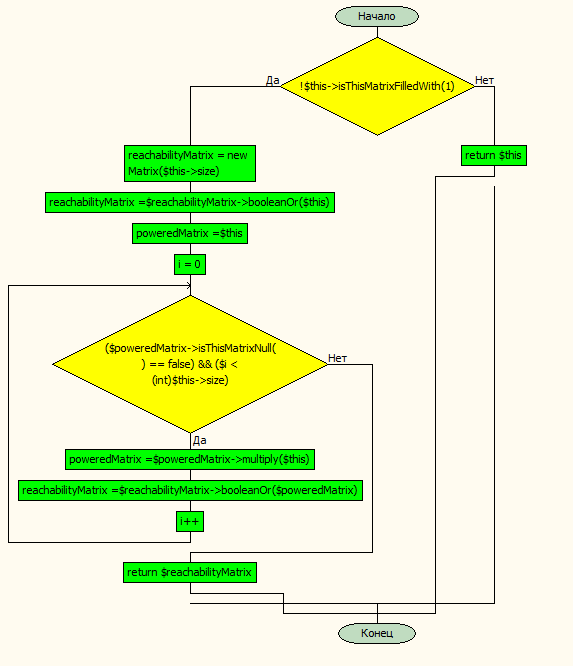
Давайте рассмотрим класс матрицы Matrix:

class Matrix  
{  
 public $matrix = [];  
 public $size;  
 public $errorCode;  
  
 public function \_\_construct($size = 0, $matrix = [])  
 {  
 if (count($matrix) > 0)  
 $this->matrix = $matrix;  
 else  
 $this->clear($this->matrix);  
 $this->size = $size;  
 }  
  
 public function loadMatrix($matrixString)  
 {  
 $matrixElements = preg\_split('/[ \n]/', $matrixString);  
 $matrixSize = sqrt(count($matrixElements));  
 if ($matrixSize - (int)$matrixSize == 0) {  
 $matrixArray = [];  
 for ($i = 0; $i < $matrixSize \* $matrixSize; $i++) {  
 $x = $i % $matrixSize;  
 $y = floor($i / $matrixSize);  
 $matrixArray[$x][$y] = $matrixElements[$i];  
 }  
 $this->size = $matrixSize;  
 $this->matrix = $matrixArray;  
 } else  
 $this->errorCode = 1;  
 }  
  
 public function multiplyMatrices($matrix1, $matrix2)  
 {  
 $resultMatrix = new Matrix($this->size);  
 for ($resultY = 0; $resultY < $this->size; $resultY++)  
 for ($resultX = 0; $resultX < $this->size; $resultX++) {  
 $resultMatrix->matrix[$resultX][$resultY] = 0;  
 for ($i = 0; $i < $this->size; $i++) {  
 $resultMatrix->matrix[$resultX][$resultY] += $matrix1[$i][$resultY] \* $matrix2[$resultX][$i];  
 }  
 }  
 return $resultMatrix;  
 }  
  
 public function multiply($anotherMatrix)  
 {  
 return $this->multiplyMatrices($this->matrix, $anotherMatrix->matrix);  
 }  
  
 public function isThisMatrixNull()  
 {  
 for ($y = 0; $y < $this->size; $y++)  
 for ($x = 0; $x < $this->size; $x++) {  
 if ($this->matrix[$x][$y] != 0)  
 return false;  
 }  
 return true;  
 }  
  
 public function isThisMatrixFilledWith($variable)  
 {  
 for ($y = 0; $y < $this->size; $y++)  
 for ($x = 0; $x < $this->size; $x++) {  
 if ($this->matrix[$x][$y] != $variable)  
 return false;  
 }  
 return true;  
 }  
  
 public function clear($variable)  
 {  
 for ($y = 0; $y < $this->size; $y++)  
 for ($x = 0; $x < $this->size; $x++) {  
 $this->matrix[$x][$y] = $variable;  
 }  
 }  
  
 public function booleanOr($anotherMatrix)  
 {  
 $resultMatrix = new Matrix($this->size);  
 for ($y = 0; $y < $this->size; $y++)  
 for ($x = 0; $x < $this->size; $x++) {  
 $val1 = $this->matrix[$x][$y];  
 $val2 = $anotherMatrix->matrix[$x][$y];  
 $val3 = (int)($val1 || $val2);  
 $resultMatrix->matrix[$x][$y] = $val3;  
 }  
 return $resultMatrix;  
 }  
  
 public function printMatrix()  
 {  
 for ($y = 0; $y < $this->size; $y++) {  
 for ($x = 0; $x < $this->size; $x++) {  
 echo $this->matrix[$x][$y] . " ";  
 }  
 echo "<br>";  
 }  
 }  
  
 public function getReachabilityMatrix()  
 {  
 if (!$this->isThisMatrixFilledWith(1)) {  
 $reachabilityMatrix = new Matrix($this->size);  
 $reachabilityMatrix = $reachabilityMatrix->booleanOr($this);  
 $poweredMatrix = $this;  
  
 $i = 0;  
 while (($poweredMatrix->isThisMatrixNull() == false) && ($i < (int)$this->size)) {  
 $poweredMatrix = $poweredMatrix->multiply($this);  
 $reachabilityMatrix = $reachabilityMatrix->booleanOr($poweredMatrix);  
 $i++;  
 }  
 return $reachabilityMatrix;  
 } else  
 return $this;  
 }  
}

Класс имеет поля matrix и size – это массив элементов матрицы и размер матрицы соответственно.

Нахождение матрицы достижимости находится в методе getReachabilityMatrix.

public function getReachabilityMatrix()  
{  
 if (!$this->isThisMatrixFilledWith(1)) {  
 $reachabilityMatrix = new Matrix($this->size);  
 $reachabilityMatrix = $reachabilityMatrix->booleanOr($this);  
 $poweredMatrix = $this;  
  
 $i = 0;  
 while (($poweredMatrix->isThisMatrixNull() == false) && ($i < (int)$this->size)) {  
 $poweredMatrix = $poweredMatrix->multiply($this);  
 $reachabilityMatrix = $reachabilityMatrix->booleanOr($poweredMatrix);  
 $i++;  
 }  
 return $reachabilityMatrix;  
 } else  
 return $this;  
}



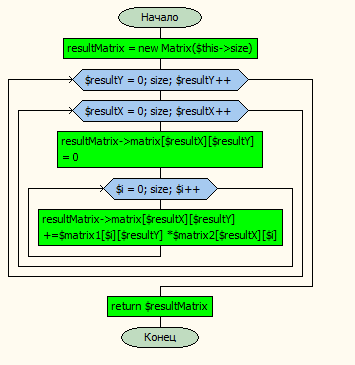
Первым делом функция проверяет, не является ли матрица полностью заполненной единицами. Если это так, то нет смысла считать матрицу достижимости, ведь и так понятно, что она просто будет полностью заполнена единицами. Затем происходит зам просчёт матрицы достижимости. Делается это с помощью возведения в степень матрицы и с каждой итерации умножения матрицы на себя производится дизъюнкция матрицы, которая в итоге станет матрицей достижимости, и матрицы, которая возводится в степень, т.е. умножается несколько раз на себя.

Для реализации этого алгоритма используются и вспомогательные функции, некоторые из которых стоит рассмотреть. Функция умножения выглядит следующим образом:

public function multiply($anotherMatrix)  
{  
 return $this->multiplyMatrices($this->matrix, $anotherMatrix->matrix);  
}

Она перемножает матрицу из аргумента с матрицей-полем класса, из которого вызывается метод multiply. Однако эта функция является посредником, само умножение же происходит в функции multiplyMatrices:

public function multiplyMatrices($matrix1, $matrix2)  
{  
 $resultMatrix = new Matrix($this->size);  
 for ($resultY = 0; $resultY < $this->size; $resultY++)  
 for ($resultX = 0; $resultX < $this->size; $resultX++) {  
 $resultMatrix->matrix[$resultX][$resultY] = 0;  
 for ($i = 0; $i < $this->size; $i++) {  
 $resultMatrix->matrix[$resultX][$resultY] += $matrix1[$i][$resultY] \* $matrix2[$resultX][$i];  
 }  
 }  
 return $resultMatrix;  
}



Тут выполняется стандартный алгоритм перемножения матриц, n-ая строка на m-ый столбец для элемента на строке n и столбце m.

**Вывод**

Данная программа протестирована и успешно работает. Программа позволяет получить матрицу достижимости из введённой матрицы. В случае некорректности ввода данных, программы оповещает об этом.

**Список источников**

Видео с youtube. Достижимость в графах. [Электронный ресурс] URL: <https://www.youtube.com/watch?v=yWBkYohi6bY&t=1395s>