Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №6

По дисциплине: «Языковые процессы интеллектуальных систем»

# Тема: «Метод поиска в глубину на графе, рекурсивная и не рекурсивная реализации алгоритма поиска в глубину»

Выполнил:

Студент 4 курса

Группы ИИ-21

Кирилович А.А.

Проверил:

Монтик Н.С.

Брест 2024

**Цель работы**: сформировать представление о методе и алгоритме поиска в глубину на графе и его применении при решении задач.

11. Разработайте программу, которая использует поиск в глубину для нахождения всех возможных путей от вершины v к вершине w. Граф представлен матрицей смежности. Программа должна выводить все найденные пути.

nonrecursive.dart

void findAllPathsNonRecursive(List<List<int>> graph, int v, int w) {

  List<List<int>> stack = [];

  stack.add([v]);

  while (stack.isNotEmpty) {

    List<int> path = stack.removeLast();

    int current = path.last;

    if (current == w) {

      print(path);

    } else {

      for (int i = 0; i < graph.length; i++) {

        if (graph[current][i] == 1 && !path.contains(i)) {

          List<int> newPath = List.from(path);

          newPath.add(i);

          stack.add(newPath);

        }

      }

    }

  }

}

void main(List<String> arguments) {

  List<List<int>> graph = [

    [0, 1, 0, 0, 1, 0, 0, 0],

    [0, 0, 1, 1, 0, 0, 0, 0],

    [0, 0, 0, 0, 0, 1, 0, 0],

    [0, 0, 0, 0, 1, 1, 1, 0],

    [0, 0, 0, 0, 0, 0, 0, 1],

    [0, 0, 0, 0, 0, 0, 1, 0],

    [0, 0, 0, 1, 0, 0, 0, 1],

    [0, 0, 0, 0, 0, 0, 0, 0],

  ];

  final int startVertex;

  final int endVertex;

  if (arguments.length == 2) {

    startVertex = int.parse(arguments[0]);

    endVertex = int.parse(arguments[1]);

  } else if (arguments.isEmpty) {

    startVertex = 0;

    endVertex = 7;

  } else {

    throw Error();

  }

  print('Все пути от вершины $startVertex до вершины $endVertex:');

  findAllPathsNonRecursive(graph, startVertex, endVertex);

}

recursive.dart

void findAllPaths(List<List<int>> graph, int v, int w) {

  List<int> path = [];

  Set<int> visited = {};

  dfs(graph, v, w, visited, path);

}

void dfs(List<List<int>> graph, int v, int w, Set<int> visited, List<int> path) {

  visited.add(v);

  path.add(v);

  if (v == w) {

    print(path);

  } else {

    for (int i = 0; i < graph.length; i++) {

      if (graph[v][i] == 1 && !visited.contains(i)) {

        dfs(graph, i, w, visited, path);

      }

    }

  }

  path.removeLast();

  visited.remove(v);

}

void main(List<String> arguments) {

  List<List<int>> graph = [

    [0, 1, 0, 0, 1, 0, 0, 0],

    [0, 0, 1, 1, 0, 0, 0, 0],

    [0, 0, 0, 0, 0, 1, 0, 0],

    [0, 0, 0, 0, 1, 1, 1, 0],

    [0, 0, 0, 0, 0, 0, 0, 1],

    [0, 0, 0, 0, 0, 0, 1, 0],

    [0, 0, 0, 1, 0, 0, 0, 1],

    [0, 0, 0, 0, 0, 0, 0, 0],

  ];

  final int startVertex;

  final int endVertex;

  if (arguments.length == 2) {

    startVertex = int.parse(arguments[0]);

    endVertex = int.parse(arguments[1]);

  } else if (arguments.isEmpty){

    startVertex = 0;

    endVertex = 7;

  } else {

    throw Error();

  }

  print('Все пути от вершины $startVertex до вершины $endVertex:');

  findAllPaths(graph, startVertex, endVertex);

}

Изображение выглядит как текст, Шрифт, снимок экрана, типография

Автоматически созданное описание

**Вывод:** в ходе выполнения лабораторной работы изучил метод и алгоритм поиска в глубину на графе.